REPORT DOCUMENTATION PAGE

AFRL-SR-AR-TR-05

0243

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
		Final Report

4. TITLE AND SUBTITLE
Cooperative Control of Multiple Unmanned Autonomous Vehicles

5. FUNDING NUMBERS
F49620-01-1-0337

6. AUTHOR(S)
Kendall E. Nygard

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Air Force Office of Scientific Research
875 North Randolph Street
Suite 325, Room 3112
Arlington, VA 22203

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
North Dakota State University
1301 12th Avenue North
Fargo, ND 58105

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION AVAILABILITY STATEMENT

Approved for public release;
distribution unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

The decision for an autonomous Unmanned Air Vehicle (VA ~9 to strike a target is an important one that has important consequences. The details regarding the decision and the process of deciding need to be well defined and understood This paper presents a Petri net approach to UA V decision making based onflizzy reasoning It shows that Petri nets are a viable toolfor modeling and validating propos itional logic for rule-based reasoning in UA V decisions. It also discusses alternative methods for modeling places as input sources for transitions representing rules.

14. SUBJECT TERMS			15. NUMBER OF PAGES
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT

Cooperative Control of Multiple Unmanned Autonomous Vehicles

Final Report

Kendall E. Nygard
Department of Computer Science and Operations Research
North Dakota State University
Fargo, ND 58105-5164

Award # F49620-01-1-0337
Air Force Office of Scientific Research

Grant organization:
North Dakota State University
1301 12th Avenue North
Fargo, ND 58105

6/3/05

Project Director
Kendall E. Nygard, Professor and Chair
Department of Computer Science and Operations Research
North Dakota State University
258 IACC
Fargo, ND 58105-5164
(701)-231-8203
Kendall.Nygard@ndsu.nodak.edu

Research Summary

The primary objectives of the research are as follows:

1. Design, implement, test, and evaluate a prototype hierarchical architecture for cooperative UAV control at the fleet, team, and individual UAV level,
2. Design, implement, test, and evaluate a market-based model for UAV team formation resource allocation,
3. Design, implement, test, and evaluate closed form discrete optimization algorithms for resource allocation in systems of UAVs.

Year 1

Four publications pertaining to the completion of the tasks required to achieve these objectives were accomplished in year 1, included in previous annual reports, and are listed among the references. In summary, with regard to objective 1, first, as reported in [Chandler et. al, 2002]and [Dargar et. al, 2002], the hierarchical control structure was extensively developed and tested. The issue of correct classification of targets was incorporated, along with closed form discrete optimization models and market-based bidding models. The role of path planning for individual UAVs is directly incorporated into a lower level in the hierarchy, and provides a source of information to support decision-making at higher levels. With regard to objective 2, a market-based model was developed, tested, and reported in [Guo et. al., 2002]. The scheme is combinatorial, and captures complex complementarities and substitutability relationships among tasks with a mediation scheme. The combinatorial approach is essential in situations with strong dependencies among tasks exist, such as when multiple targets are in close proximity to one another. Agent UAVs are allowed to sell and buy tasks simultaneously, and a trading mechanism that extends one-side combinatorial auctions is employed. Finally, we developed an algorithm for efficiently generating small subsets of feasible trades that are likely to be attractive. For objective 3, a dynamic network flow optimization model for allocating UAVs resources was devised and is reported in [Nygard et. al., 2002]. This model captures distinct types of tasks, such as searching, classifying, striking, and assessing damage to targets. By employing a fast network solver, the model is capable of near-real solutions, allowing UAVs controlled by the model to adapt to changing conditions in the battlefield.

Year 2

In year two of the project, progress on objective 1 was made in several ways. First, as reported in [Altenburg, et. al, 2002], an aspect of the framework that utilizes low-level emergent swarm intelligence was developed. It was demonstrated that local behaviors governed by a state machine could execute a structured global search strategy. Although biological swarms inspire the approach, the work establishes that the behaviors can be specified in such a way that uncontrolled chaotic behavior is avoided. With regard to objective 2, the market-based model was extended to directly address multiple task allocation with subteam formation. [Guo et. al., 2002]. The combinatorics of the problem are dealt with by using an up front heuristic depth-first search to generate promising tasks for each agent. This significantly reduces the computational burden of the cost calculations, yet provides near-optimal allocations. There is almost a complete lack of quantitative models that directly address subteam formation, so this aspect of the market-based model significantly extends available methodologies that can be applied to highest end of the control hierarchy for UAV mission control. Under objective 3, the dynamic network flow optimization model reported in September of 2002 was used as the basis and direct competitor in a neuro-dynamic programming approach to fast task allocation under dynamically changing battlefield conditions

(e.g., pop-up threats, opportunistic targets) [Anwar et. al., 2002]. In essence, a neural network is used as an approximation of the cost-to-go function of dynamic programming, providing a very fast means of generating high-performance task allocations on the fly. In addition, a more comprehensive integer programming model was developed, for use in determining optimal roles for individual UAVs to play in team-oriented missions [Hennebry et. al., 2004]. Although not usable for dynamic reallocations, the model is expected to be useful in generating very high performance task allocations in situations where the opportunities and threats are known somewhat in advance. The model is also useful as a benchmark for the fast heuristic procedures under development.

Year 3

In the final year of the project, the reactive behavior approach was extended to support an asset patrolling scenario and is attached to this paper and reported in [Nygard, 2004]. This is a continuation of the work involved with completing objective 1. The work is inspired by the need for persistent patrolling operations in recent conflicts of the world, including operations in Iraq. Also supporting objective 1, a Partially-Observable Markovian Decision Process (POMDP) methodology was developed to explicitly provide a modeling framework for choosing alternative available mission decisions under conditions of incomplete information. This is reported in [Schesvold, 2003]. Finally, [Lundell 2005] reports on the use of a promising alternative model for cooperative control based on Petri net modeling.

The project has involved over graduate students Kenneth Grigsby, Martin Lundell, Benzir Ahmed, Wenge Guo, Haiyan Qaio, Md. Anwar, and Chin Lua. Reseasrch Associates Michael Hennebry, Doug Schesvold, and Jingpeng Tang were supported. One graduate student supported spent the summer of 2002 at the Air Force Research Laboratory at Wright-Patterson AFB to help keep communication ongoing with that group of scientists, and to help the project stay focused on the appropriate Air Force needs. Dr. Nygard continues to build solid relationships with Air Force scientists and investigators such as Dr. Siva Banda and Phillip Chandler at the Air Force Research Laboratories at Wright-Patterson Air Force Base and visit AFRL laboratories to share results and continue collaborations.

List of Publications

Publications pertaining to year 1.

[Altenburg 2002] Altenburg, Karl, Joseph Schlecht, and Kendall E. Nygard, "An Agent-based Simulation for Modeling Intelligent Munitions", Proceedings of the WSEAS conference, Skiathos, Greece, September, 2002

[Guo 2002] Guo, W., K. Nygard, H. Qiao and A. Kamel (2002). Multiple Task Allocation Problems with Team Formation. ISCA 11th International Conference on Intelligent Systems, Boston, MA, International Society for Computers and Their Applications (ISCA).

[Anwar 2002] Anwar., M. Kamel, A. and Kendall E. Nygard, Task Allocation to Unmanned Aerial Vehicles in A Dynamic Environment by Neuro-Dynamic Programming, Technical Report, Department of Compute Science and Operations Research, 2002.

[Hennebry 2002] Hennebry, Michael, Ahmed Kamel, and Kendall E. Nygard, An Integer Progamming Model for Assigning Unmanned Air Vehicles to Tasks, in Recent Developments in Cooperative Control and Optimization, Kluwer publishing, Sergei Butenko and Robert Murphy, Eds, forthcoming, 2004.

Publications pertaining to year 2.

[Altenburg 2002] Altenburg, Karl, Joseph Schlecht, and Kendall E. Nygard, "An Agent-based Simulation for Modeling Intelligent Munitions", Proceedings of the WSEAS conference, Skiathos, Greece, September, 2002

[Guo 2002] Guo, W., K. Nygard, H. Qiao and A. Kamel (2002). Multiple Task Allocation Problems with Team Formation. ISCA 11th International Conference on Intelligent Systems, Boston, MA, International Society for Computers and Their Applications (ISCA).

[Anwar 2002] Anwar., M. Kamel, A. and Kendall E. Nygard, Task Allocation to Unmanned Aerial Vehicles in A Dynamic Environment by Neuro-Dynamic Programming, Technical Report, Department of Compute Science and Operations Research, 2002.

[Hennebry 2002] Hennebry, Michael, Ahmed Kamel, and Kendall E. Nygard, An Integer Progamming Model for Assigning Unmanned Air Vehicles to Tasks, in Recent Developments in Cooperative Control and Optimization, Kluwer publishing, Sergei Butenko and Robert Murphy, Eds, forthcoming, 2004.

Publications pertaining to final year.

[Lundell 2005], Lundell, Martin, Jingpeng Tang, and Kendall E. Nygard, Fuzzy Petri Net for UAV Decision Making, in The 2005 International Symposium on Collaborative Technologies and Systems, May 15-20, 2005, Saint Louis, Missouri, USA.

[Nygard 2004], Nygard, Kendall E., Altenburg, K., Tang, K. and D. Schesvold, A Decentralized Swarm Approach to Asset Patrolling with Unmanned Air Vehicles, in Theory and Algorithms for Coopertive Systems, Kluwer publishing, Series on Computers And Operations Research, Volume 4, Don Grundel, Robert Murphy, and Panos M. Pardalos, Eds, 2004.

[Schesvold 2003] Schesvold, Doug., Tang, Jingpeng.,Ahmed, Benzir Md., Altenburg, Karl, and Nygard, Kendall E., POMDP Planning for High Level UAV Decisions: Search vs. Strike in Proceedings of the 16th International Conference on Computer Applications in Industry and Engineering (CAINE_03), Las Vegas, November, 2003.

Fuzzy Petri Net for UAV Decision Making

Martin Lundell
University of Minnesota, Crookston
mlundell@umn.edu

Jingpeng Tang
University of Minnesota, Crookston
jptang@umn.edu

Kendall Nygard
North Dakota State University
Kendall.Nygard@ndsu.edu

ABSTRACT

The decision for an autonomous Unmanned Air Vehicle (UAV) to strike a target is an important one that has important consequences. The details regarding the decision and the process of deciding need to be well defined and understood. This paper presents a Petri net approach to UAV decision making based on fuzzy reasoning. It shows that Petri nets are a viable tool for modeling and validating propositional logic for rule-based reasoning in UAV decisions. It also discusses alternative methods for modeling places as input sources for transitions representing rules.

KEYWORDS— *Petri net, Fuzzy Reasoning, UAV, control structure, modeling*

1. INTRODUCTION.

Unmanned Air Vehicles (UAVs) have been, and are currently, receiving much attention in research. To date, much of the practical applications of UAVs have been for reconnaissance with their decision and control being handled remotely. Current research is examining the possibilities of UAVs as autonomous agents working individually or collaboratively with other autonomous UAVs [6][7][8]. One desirable attribute for an autonomous UAV is the ability to determine the value of an enemy target, the level of threat that target imposes, and whether or not it should strike that target. One potential model that can be employed in the decision making process is that of fuzzy reasoning.

Since the information that a UAV can have about a target at any point in time may not be complete, the UAV could use rules-based reasoning to compute the confidence it has in several factors before making a decision to strike or to search for a new target.

2. FUZZY PETRI NET.

Petri nets [1] are a graphical and mathematical modeling tool. They can be used as a communication tool similar to other modeling notations like state-transition diagrams and entity-relationship diagrams. As a mathematical tool it can set up mathematical models that govern the behavior of systems. Those systems can be asynchronous, concurrent, parallel, stochastic, or non-deterministic.

Using a Petri net formalism allows us two primary advantages [3]. It allows us to visualize the structure of the rules-based system, making the model more legible and easier to understand; and secondly to express the behavior of the system in mathematical forms.

The term Fuzzy Petri Net (FPN) has been used to describe Petri nets that use their formalism to implement fuzzy reasoning algorithms. In FPNs, places can represent propositions, transitions can represent rules, and tokens can represent truth values. In [3], the authors propose an extension called Fuzzy Reasoning Petri Nets (FRPNs) where the properties of the Petri net are further defined: 1) If a place represents a truth degree, it can have, at most, one token. 2) FRPNs are conflict-free nets as rules may share propositions. 3) Tokens are not removed from the input places after it fires. And 4) Complementary arcs do not inhibit the firing of a transition if its place has a token.

The remainder of this document is organized as follows. In section 3 we will define some variables and concepts. In section 4 we will apply a fuzzy Petri net to a UAV decision making scenario. In section 5 we will discuss the implications of using fuzzy Petri nets in such rule based reasoning. Section 6 states the conclusions and section 7 lays out some future work.

3. DEFINITIONS.

- TGV be the linguistic variable for Target Value

representing a number in the range [0, 100].

- Let THV be the linguistic variable for Threat Value representing a number in the range [0, 100].
- Let FLV be the linguistic variable for Fuel Level Value representing a number in the range [0, 100].
- Let R1, R2, ... Rn be n-rules that represent the fussy reasoning scenario.

In this paper we use Mamdani's fuzzy implication operators for triangular norms and triangular conorms where,

minimum/maximum:

$$MIN(a, b) = \min\{a, b\} = a \wedge b, \quad (1)$$

$$MAX(a, b) = \max\{a, b\} = a \vee b$$

4. FUZZY PETRI NET MODEL FOR UAV.

In the following scenario we assume a UAV will be using its sensors to determine a target value, the threat of a target, and its own fuel level. Each UAV is terminal in the sense that it will search for a target and in the processing of striking a target, it too will be destroyed. If it consumes all it's fuel before striking a target, the UAV will crash. Thus, it is better that a UAV strike a target with a low value than none at all. The following scenario is a simplified one to enhance clarity of the process. There could be more properties taken into account. For example, sensors could detect radar or radio communications from a potential target and use that information in its decision making process and others we will discuss shortly.

The process for implementing the scenario will be A) fuzzification of input values, B) rule evaluation, C) aggregation of rule outputs, and D) defuzzification

4.1 Fuzzification of input values.

Crisp Input:
Let the TVG be 55.
Let the THV be 40.
Let the FLV be 80.

The following equations are used to determine the degree of participation of a value in a fuzzy set. For simplicity's sake, all three membership functions have the same plotting values along the x axis and the same fuzzy set allocation into high, moderate, and low. In a practical application the military would decide the membership function of their fuzzy sets.

$$low(v) = \begin{cases} 1 & if\ v \le 30 \\ 1-(v-30)/20 & if\ 30 \le v \le 50 \\ 0 & otherwise \end{cases} \quad (2)$$

$$moderate(v) = \begin{cases} 1-|v-50|/20 & if\ 30 \le v \le 70 \\ 0 & otherwise \end{cases} \quad (3)$$

$$high(v) = \begin{cases} 1 & if\ v \ge 70 \\ 1-(v-30)/20 & if\ 50 \le v \le 70 \\ 0 & otherwise \end{cases} \quad (4)$$

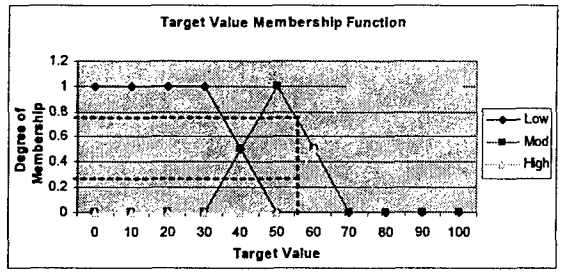

FIGURE 1a. Target Value Membership (TGV)

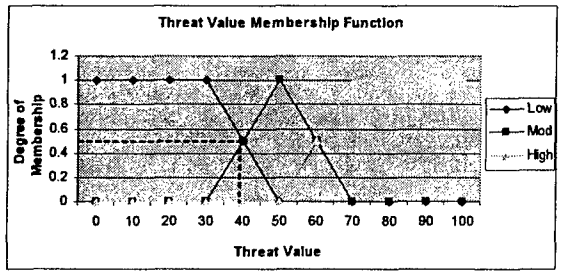

FIGURE 1b. Threat Value Membership (THV)

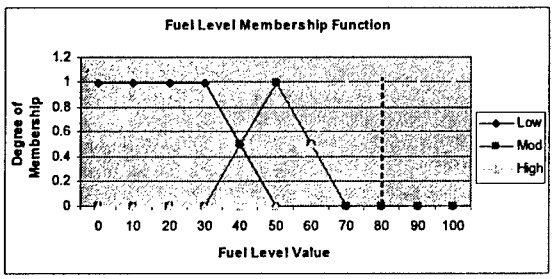

FIGURE 1c. Fuel Level Membership (FLV)

Applying the crisp input value for TGV to the membership function where v=TGV, we get the values for the TGV's participation in the low, moderate, and high fuzzy sets as low = 0, moderate = 0.75, and high = 0.25. This is represented in fig. 1a. We proceed by applying the membership functions to the crisp values for THV and FLV. For THV the participation in low = 0.5, moderate = 0.5 and high = 0. This is represented in fig. 1b. For FLV the participation in low = 0, moderate = 0, and high = 1. This is represented in fig. 1c.

4.2. Rule evaluation.

We have identified the following rules for this scenario:

R1: If THV is high then strike target.
R2: If TGV is high then strike target.
R3: If FLV is low then strike target.
R4: If TGV is moderate and THV is moderate and FLV is moderate then strike target.
R5: IF TGV is low and THV is low then continue search.
R6: If TGV is moderate and THV is low and FLV is moderate then strike target.
R7: If TGV is low and THV is moderate and FLV is moderate then strike target.
R8: If TGV is moderate and THV is moderate and FLV is high then continue search.
R9: If TGV is moderate and THV is low and FLV is high then continue search.
R10: If TGV is low and THV is moderate and FLV is high then continue search.

Using Mamdani's implication operators we take the minimum truth value in a rule antecedent and apply that value as the degree of participation the consequent has in its fuzzy set. This is represented in step 2 of fig 3.

4.3. Aggregation of rule outputs

In our scenario the consequent member function, Decision, has two fuzzy sets: Search and Strike. This is shown in fig 2.

In aggregating the rule outputs we again use Mamdani's implication operators, this time using the max function to determine the maximum degree of participation in each of the consequent fuzzy sets. In other words, we want the max value for Strike that was output by all the rules where Strike was a consequent fuzzy set and we want the max value for Search that was output by all the rules where Search was a consequent fuzzy set.

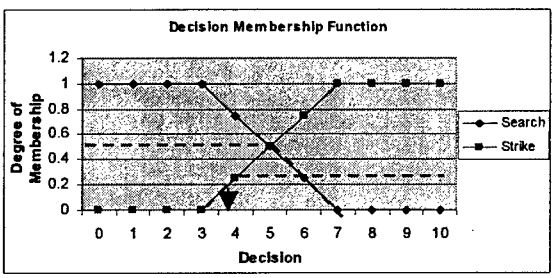

FIGURE 2 Decision Membership

4.4. Defuzzification

After finding the max value from all rule consequents for each fuzzy set we need to turn those values into a crisp output value. We chose a centroid technique that finds the center of gravity (COG) from the aggregate fuzzy sets resulting from step 3 in fig 3. The center of gravity can be expressed as:

$$COG = \frac{\int_a^b \mu_A(x)\, x\, dx}{\int_a^b \mu_A(x)\, dx} \qquad (5)$$

Using the data from our Petri net in fig. 3, our COG = 3.83.

$$COG = \frac{(0+1+2+3)\times 0.5 + (7+8+9+10)\times 0.25}{0.5+0.5+0.5+0.5+0.25+0.25+0.25+0.25} = 3.83$$

The crisp output of 3.83 would indicate the UAV's

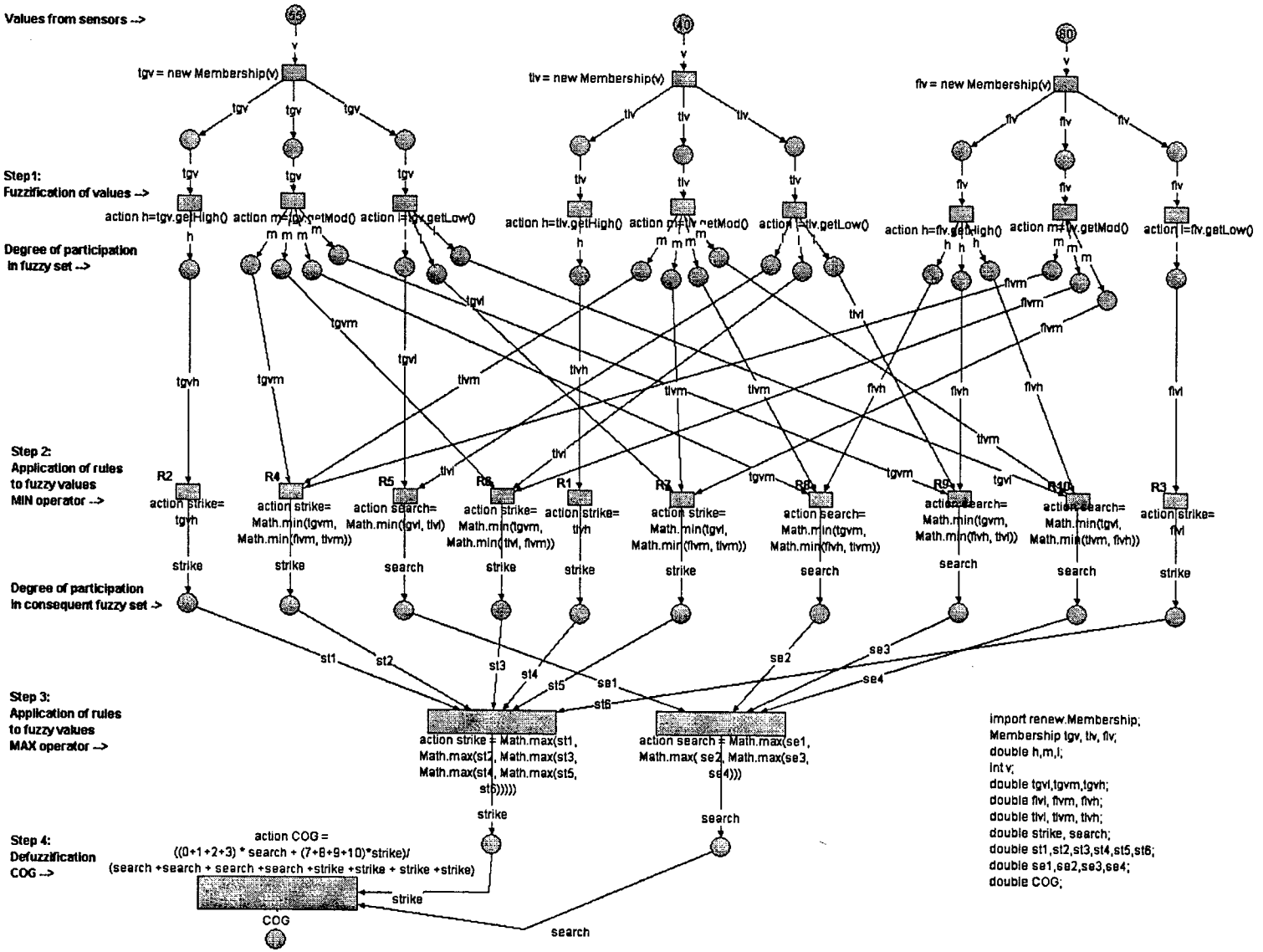

FIGURE 3. Fuzzy Petri net for UAV decision

decision to Search.

5. DISCUSSION.

5.1. On Petri Net Modeling

Using the FRPN model proposed in [3], the tokens in places do not get consumed after a token (truth value) is used in a transition (rule). The authors in [3] argue that a conflict would arise if the truth value were to be removed (i.e. other rules would not be able access the truth value). In [9] the author suggests a return arc to put the token back in it's originating place after the transition fires. This is diagrammed in fig 4a. As noted in [3], this solves the conflict problem but weakens the parallelism of the Petri net.

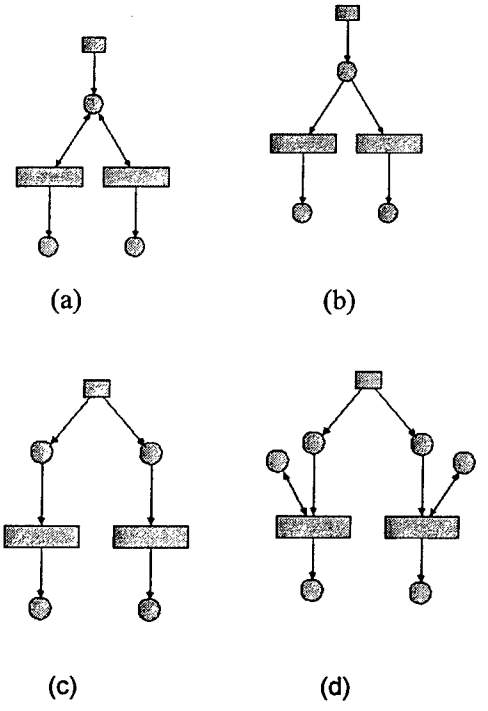

FIGURE 4. Net Structure

In [3] the authors propose the solution in fig. 4b where the token is used but is assumed to not be consumed. While this solves the conflict issue and allows for parallel firing of transitions it poses other problems. This assumption, if not clearly stated, makes it easy to misinterpret the Petri net. Also, if the token is not consumed, and the place is assumed to hold a single token representing the participation in a fuzzy set, the source transition would not be able to place a new token in that place. This is significant if there exists multiple reasoning sequences through this net.

We propose that a separate place be generated for every transition that will consume a given token. See Fig. 4c. Each place would then have one outgoing arc to its corresponding rule. This preserves the parallelism of the net and also resolves the conflict issue mentioned earlier. It also holds to more of the traditional rules of Petri nets. One downside to this method is a proliferation of places with the same value. This may make the Petri Net more difficult to read.

The net structure in fig. 4d would allow for comparison of values from one iteration of the reasoning sequence to the next. Consider the following expansion of the UAV Scenario. A new input value of "distance to the target" is added to the equation. This would allow for a third fuzzy set to be added to the consequent member function "Decision". Let the this new fuzzy set be described as "Continue Evaluation". Rules that could be added may look like this:

R1: If distance is high continue evaluation.
R2: If distance is moderate and threat is high strike target.
R3: If distance is low and target value is low and fuel level is moderate continue search.
...

Under these circumstances we may want to determine whether a target value was increasing or decreasing as we approached the target, and apply rules based on this information. We may do repeated evaluations of the same target (if we had not chosen to strike the target immediately). It would be valuable to have the truth values consumed by rules and have the arc be disabled until the next sensor reading is fuzzified. Since it is apparent that comparing new truth values to previous ones may be beneficial, the structure in fig. 4d would place the previous truth value in a separate place, and consume it a second time for comparison to the newly generated truth value.

5.2. On the UAV Petri Net process

It is vital that all possible paths in a fuzzy reasoning tree are realized in a decision making model for UAV target strikes. Using a fuzzy Petri net allows us to validate all paths. It also allows us to validate our fuzzy reasoning rules. We can provide sample crisp

input and traverse our Petri Net, realizing each rule and producing a final, defuzzified output. This output can be evaluated against expected decisions. Autonomous UAV's that rely on pre-flight and/or sensory information to make decisions are a type of knowledge based system. Modeling this system as a FPN allows us to detect inconsistencies in the system. Inconsistencies exist where there are conflicting, dead-end, or redundant if condition rules[2]. The process of modeling these rules in a Petri net formalism allows us to validate those rules for completeness, preciseness and consistency.

The development of fuzzy reasoning rules for implementation in software is an exercise in requirements engineering. Revealing inconsistencies in rules early in the development of a system allows for savings in development time and overall project cost. Complete and consistent rules that have been validated gives us a higher level of confidence that the developer writing the software will have the correct requirements and offers a model to validate their code upon completion.

6. CONCLUTION.

For many years researchers have looked at applying Petri nets to validate Rule Based Systems. More recently the idea of fuzzy Petri nets emerged and have been used to validate fuzzy reasoning trees. The main intent of this exercise was to apply fuzzy reasoning and Petri nets to UAV decisions. In a simple scenario, it seems to be a viable option and it will be interesting to implement it on a larger scale. In our experience, using the fuzzy Petri net proved valuable in validating our rules for completeness, preciseness, and consistency as we took input values and traversed the Petri net to came up with a final decision which we could determine as a reasonable or unreasonable decision.

We also conclude that fuzzy Petri nets can be implemented without changing the traditional Petri net's rules about conflict, consumption of tokens, arc enablement, and token placement.

7. FUTURE WORK.

Implementation of the fuzzy reasoning will be done in a simulation environment and its accuracy and efficiency will be tested against other decision making models (i.e. Bayesian Decision Analysis, Rough Set). We are also interested in looking at the practicality and feasibility of organizing fuzzy reasoning rules into a hierarchy where the consequent of one rule may be the antecedent of another rule and applying hierarchical fuzzy Petri nets to that scenario.

REFERENCES

[1] Murata, T., "Petri Nets: Properties, Analysis and Applications." *Proceedings of the IEEE.*, Vol. 77, No. 4, April 1989.

[2] Koriem, S. M., "A Fuzzy Petri Net Tool for Modeling and Verification of Knowledge-Based Systems," *Comput. J.*, vol. 43, no. 3, pp. 206–223, 2000.

[3] Gao, M., M. Zhou, X. Huang, Z. Wu, "Fuzzy Reasoning Petri Nets", *IEEE Trans. On Systems Man and Cybernetics-Part A: Systems and Humans*, Vol. 33, No. 3, May 2003, pp. 314-324.

[4] Gao, M., M.Zhou, Y.Tang, "Intelligent Decision Making in Disassembly Process Based on Fuzzy Reasoning Petri Nets", *IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics*, Vol. 34, No. 5. October 2004. pp 2029-2034

[5] Looney, C. G., "Fuzzy Petri nets for rule-based decisionmaking", *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 18 , Issue: 1 , Jan.-Feb. 1988, pp. 178 - 183

[6] Altenburg, K., J. Schlecht, and K. Nygard. "An Agent-based Simulation for Modeling Intelligent Munitions," *In Proc. of the Second WSEAS Int. Conf. on Simulation, Modeling and Optimization*, Skiathos, Greece, 2002.

[7] Lua, C., K. Altenburg, and K. Nygard. "Synchronized Multi-Point Attack by Autonomous Reactive Vehicles with Simple Local Communication." *In Proc. of the IEEE Swarm Intelligence Symposium*, Indianapolis, Indiana, 2003.

[8] Joseph, J. S., K. Altenburg, B. Ahmed, and K. Nygard. Decentralized Search by Unmanned Air Vehicles using Local Communication. *In Proc. of the International Conference on Artificial Intelligence*, Volume II, Las Vegas, pp. 757-762, 2003.

[9] Nazareth, D. L., "Investigating the Application of Petri Nets for Rule-Based System Verification," *IEEE Trans. Know. Data Eng.*, Vol. 4, March 1993, pp402-415.

Chapter 1

A DECENTRALIZED SWARM APPROACH TO ASSET PATROLLING WITH UNMANNED AIR VEHICLES

Kendall E. Nygard
Department of Computer Science and Operations Research
North Dakota State University
Fargo, ND 58105-5164
Kendall.Nygard@ndsu.nodak.edu

Karl Altenburg
Department of Computer Science and Operations Research
North Dakota State University
Fargo, ND 58105-5164
Karl.Altenburg@ndsu.nodak.edu

Jingpeng Tang
Department of Computer Science and Operations Research
North Dakota State University
Fargo, ND 58105-5164
Jingpeng.Tang@ndsu.nodak.edu

Doug Schesvold
Department of Computer Science and Operations Research
North Dakota State University
Fargo, ND 58105-5164
schesvo@web.cs.ndsu.nodak.edu

Abstract We present a procedure for controlling a team of Unmanned Air Vehicles (UAVs) for establishing patrol patterns to protect an asset on the ground. The control is decentralized and follows a reactive, behavior-based, emergent intelligent swarm design. The patrol patterns consist of flight tracks with different radii and altitudes around the asset. The multiple tracks help maintain a persistent presence around the asset for the purposes of surveillance and the destruction of hostile intruders. Populating inner tracks is favored over outer tracks, and is accomplished through behaviors that comprise a track switching protocol. Collision avoidance is maintained. Global communication is assumed to be unavailable, and control is established only through passive sensors and minimal short-range radio communication. The model is implemented and successfully demonstrated in an agent-based, simulated urban environment. The simulation establishes that the emergent, behavior-based patrol procedure for UAVs is effective, robust, and scalable. The approach is especially well suited for numerous, small, inexpensive, and expendable UAVs.

Keywords: swarm, emergent intelligence, decentralized control, patrol

1. Introduction

A bottom-up approach to decentralized control of Unmanned Air Vehicles (UAVs) is investigated in this research. The purpose of the research is to develop a model for emergent formation of UAVs into functional teams that cooperatively complete a mission, such as cooperatively patrolling an asset of high interest and striking any moving or static hostile intruders. Emergent team formation involves the creation of teams without centralized control and based on individual decisions and local information. Most previous UAV mission planning and cooperative control employ global optimization techniques which assume perfect (or near perfect) global communication and complete knowledge sharing. Since reliable global communication in threatening situations is not realistic, systems that rely on it are prone to failure. Other failures that adversely affect global optimization techniques include: loss of global positioning, communication network saturation, lack of battlefield intelligence, highly dynamic battlefield conditions, and the presence of many UAVs within the operational environment [1]. Our simulation shows that mission objectives can be accomplished if all agents follow the same protocols even in the absence of inter-agent communication. The philosophy guiding this research is that of emergent intelligence and its emphasis on bottom-up, decentralized, and behavior-based control. We believe bottom-up approaches are more robust than globally optimized approaches with respect to individual tasks in uncertain and

dynamic environments. The emergent intelligence approach relies little on a priori, situational knowledge or high-bandwidth, inter-agent communication. Solutions derived from emergent intelligence are highly adaptive in complex, dynamic, and uncertain environments and they offer a flexibility not easily attained by rigid, globally optimized solutions that assume perfect communication.

2. Simulation Framework

The research builds upon a previously developed agent-based framework to simulate UAVs as virtual agents [2]. This framework is known as ASAS - Autonomous Search and Attack System. By extending the generic, object-oriented agent structure in the framework, we created UAV agents with the intent of simulating the characteristics of small, low cost, expendable UAVs. In an effort to obtain a reasonably high-fidelity model, we assume that agents have limited capabilities. These limitations extend to the UAV's computational processing power, memory, and communication capability. It is also assumed that a UAV's sensors, actuators, and control systems are subject to noise and failures.

Individual UAV agents rely on local information obtained from its sensors and process that information locally. An agent has little or no dependence on another agent's state or presence. However, the agents are opportunistic: if information about another agent is available, that information may be used. Simple signal transmitters and sensors are attached to the agents to allow for limited range, broadcast or directional communication for opportunistic cooperation between agents. Signal reception is often limited to an agent's nearest neighbor. Therefore, an agent may be unaware that its cooperation with a close neighbor may propagate and result in team formation; teams are an epiphenomena of individual agent behavior.

Individual UAV agents are physically simulated with simple actuators allowing for turning (a virtual, coordinated roll and bank), and acceleration based on a simple, discrete set of velocities: slow, cruise, and fast. These capabilities allow the agents to model the rudimentary functionality of operational UAVs.

The control philosophy for the agents is based on task achieving modules with tight sensor-actuator coupling. Providing for the persistence of behavior in the absence of a triggering sensation requires some state information. The agents employ discrete states and may act differently to similar sensations in different states.

3. Asset Patrol Mission

Many situations may arise where it is deemed necessary to protect vital assets in high threat environments. An example of this is in the area of homeland security in which intelligence indicates that particular assets could be at risk from terrorist attacks. Mission goals for protecting such assets may include maintenance of a persistent presence around the asset, surveillance, and destruction of hostile intruders. A UAV is an ideal choice for carrying out this type of mission. A persistent presence around the asset could be maintained by establishing flight patrol patterns around the asset. Multiple UAVs in these patrol patterns at any point in time would ensure complete surveillance coverage and provide redundancy that would minimize the impact of individual UAV failures in the overall mission objective.

4. Asset Patrol Algorithm

4.1. Patrol Structure

The patrol patterns consist of flight tracks with different radii and altitudes around the asset. The multiple tracks help maintain a persistent presence around the asset for the purposes of surveillance and the destruction of hostile intruders. They also provide multiple viewpoints for surveillance as well as multiple layers of protection. Populating inner tracks is favored over outer tracks and is accomplished through behaviors that comprise a track switching protocol. Collisions in an urban area, especially around the asset being protected, would be extremely hazardous. Therefore, one of the main objectives of the protocols is that of collision avoidance. The altitude of the patrol tracks is proportional to the radius - lower tracks are smaller. Each patrol track consists of a fixed number of waypoints that form a regular polygon with the asset at its center.

4.2. UAV Sensor/Communication Capability

Global communication is assumed to be unavailable, and control is established only through passive sensors and minimal, short-range radio communication. One of the main objectives in the design philosophy is to determine what can be accomplished with minimal inter-agent communication. The motivation for this is to build systems that are highly robust. Systems that do not rely on capabilities that are prone to failure, such as global communication, are inherently more robust. Greater communication capabilities may be considered later to increase performance. The advantage of our design philosophy is that if these added

communication capabilities fail, the system will still function reliably because it was designed to work without them.

4.3. UAV Behaviors

The high level objective of asset patrol is accomplished (emerges) from the more local UAV behaviors of *collision avoidance, patrolling*, and *attacking*. The *collision avoidance* and *attacking* behaviors are similar to those used in the sweep search described in [1]. The focus here is on the *patrolling* behavior.

The high-level control structure is illustrated in the state chart of Figure 1.1. The control is hierarchical, with the *Choose* module of Figure 1.1 being a high-level construct charged with identifying which lower-level state chart should appropriately be in control in the current situation. The current situation is assessed at regular time intervals by the *Choose* module. At each cycle, sensory input is processed to determine the best choice of action. Figure 1.2 illustrates an expansion of the *Patrol Asset* module into its lower-level state chart consisting of the behaviors *enter patrol, patrol, seek gap*, and *exit patrol.*

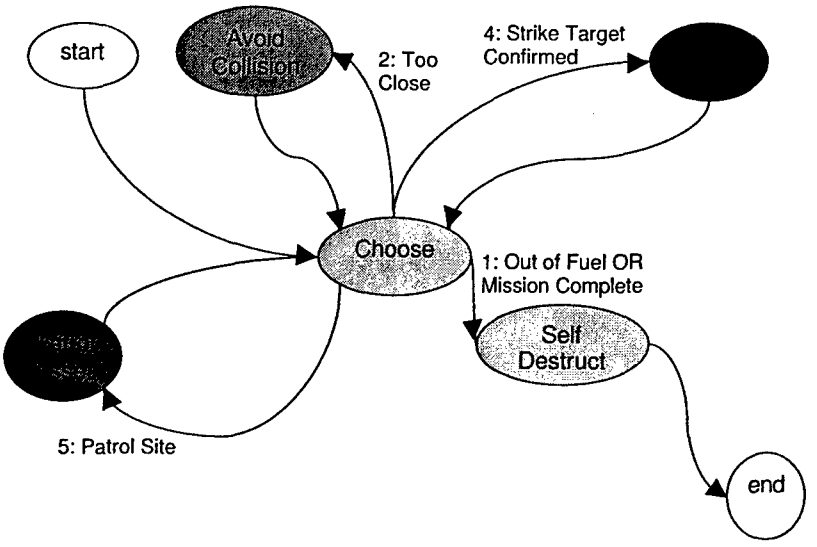

Figure 1.1: Hierarchical State charts of UAV behavior.

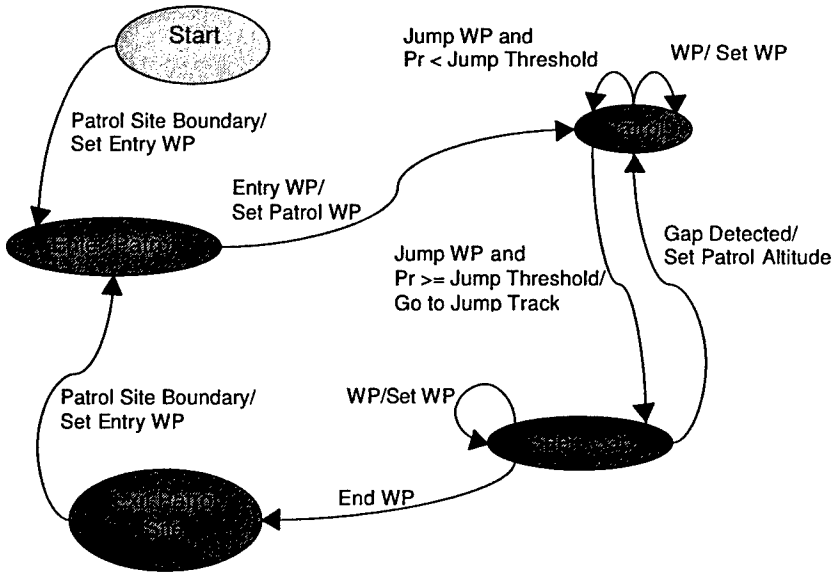

Figure 1.2: Detailed state charts of UAV patrol asset behavior.

4.4. Enter Patrol Behavior

The enter patrol behavior is executed when the UAV is attempting to enter the outermost patrol track. When the UAV reaches a particular distance from the asset, it maneuvers to orient itself in the direction of patrol flight. This patrol direction is known in advance and is either clockwise or counterclockwise. Once the UAV is oriented in the patrol direction, it calculates which of the pre-specified entry points of the outer track is the closest to its current heading. If the UAV doesn't encounter any obstacles, such as other UAVs, on its way to the entry point, it will enter the outer patrol track. If another UAV is encountered, it will fly away from the asset for some distance before repeating the enter patrol behavior. The behavior of flying away from the asset when encountering other UAVs in close proximity provides congestion control for the outer track.

4.5. Patrol Behavior

The patrol behavior consists of orbiting around the asset in the current track by flying from waypoint to waypoint while scanning for possible intruders. UAVs maintain cruise speed while in a patrol track. While

patrolling in an outer track, a UAV use a probability calculation to decide wether to attempt to switch to the next inner track. This decision is always made at a pre-specified waypoint. Limiting track switching attempts to a pre-specified point minimizes potential collisions.

4.6. Track Switching Protocol

The decision to switch tracks is based on the UAV's perception of congestion of the target track. If a UAV tries unsuccessfully to switch to a particular track, it will remember this and lower its probability of trying the next time. Initially, the UAVs attempt track switches with 100% probability. A track switch attempt consists of three steps, as depicted in Figure 1.3: 1) Jump from patrol track to jump track at the pre-specified waypoint, 2) Jump from jump track to patrol track if a gap is detected, and 3) Start over if a gap is not detected before the pre-specified exit point is reached.

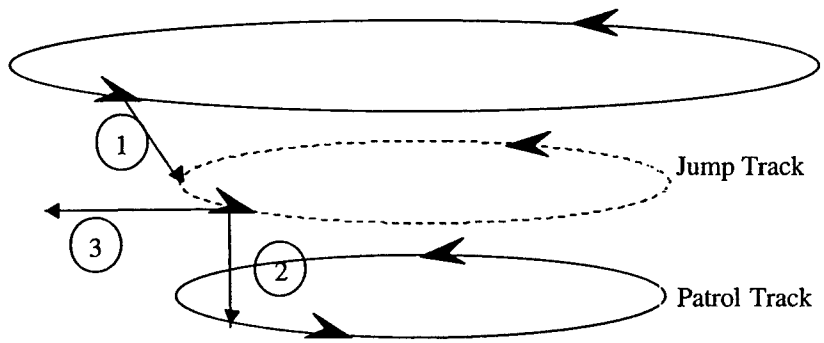

Figure 1.3: Track switch protocol.

The jump track for a particular patrol track is the same radius as the desired patrol track but at an altitude that is half way between the two tracks involved in the switch. Once the UAV enters the jump track at the pre-specified point, it executes the *gap seeking* behavior.

4.7. Gap Seeking Behavior

After entering the jump track, the UAV accelerates to fast speed and begins looking for a point where it can fit in the desired patrol track . This activity is known as the gap seeking behavior. The point that the UAV seeks is such that a minimum separation distance between UAVs

is maintained. It is assumed that the UAV has only forward scanning visual sensors. A UAV in the jump track uses a timer to determine if there is enough room behind it in the target patrol track. This timer is set to zero when the UAV first enters the jump track. Each time the jump UAV observes a patrol UAV directly below it, the timer is reset to zero. Given the difference in speed of UAVs in the jump track and UAVs in the patrol track, the jump UAV determines that there is enough room behind when the timer reaches a certain value. If the timer reaches this value, the jump UAV simply scans to see if there is enough room ahead as well. If so, the UAV has found a gap. The timer reset is illustrated in Figure 1.4. The gap calculation is illustrated in Figure 1.5.

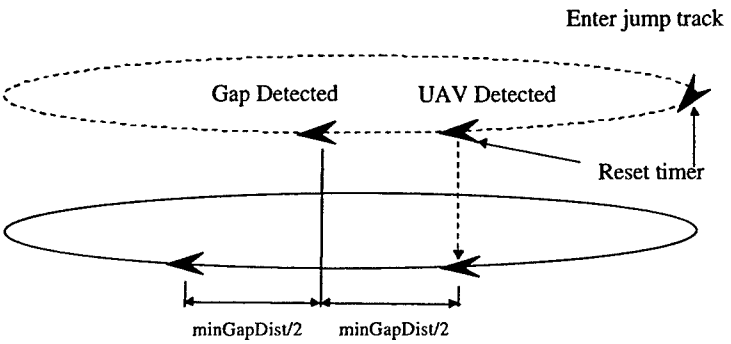

Figure 1.4: Gap detection.

$$t(v_{fast} - v_{cruise}) \geq \frac{minGapDist}{2} \tag{1}$$

$$t_{fast} = \frac{\Delta l + \frac{minGapDist}{2}}{v_{fast} - v_{cruise}} \tag{2}$$

If the UAV doesn't find a gap large enough before it reaches the exit point, it will exit the patrol area by executing the exit patrol behavior. The entry and exit points of the jump track are such that a UAV is in the jump track slightly less than one complete orbit. This restriction eliminates possible collisions in the jump track.

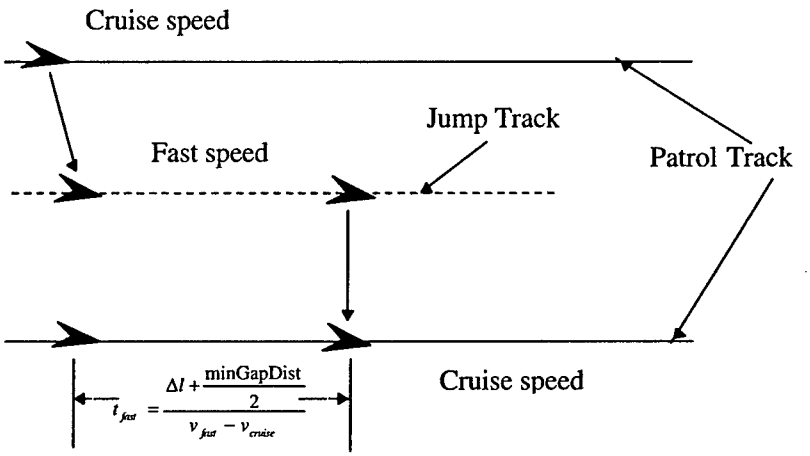

Figure 1.5: Gap timer calculation.

4.8. Exit Patrol Behavior

The purpose of the exit patrol behavior is to exit the patrol area after a failed track switch to avoid collisions with other patrolling UAVs. Starting at the exit point of the jump track, the UAV flies away from the asset until it is well beyond the outer most patrol track. Until it reaches this point, it maintains the altitude of the jump track it just exited since no other patrolling UAVs are at this altitude. It then begins climbing to the altitude of the outer most patrol track. Then the enter patrol behavior is invoked. The exit patrol behavior may also be used when UAVs are low on fuel and need to return to base.

5. Experimental Results and Observations

The system has been tested under varying experimental conditions which include: varying number of UAVs, varying number of threats, and differing track shapes. Threat density was varied from none, low, and high with 0, 5 to 10, and 10 to 15 threats respectively. The experimental results with varying numbers of UAVs and threats using hexagonal tracks are shown in Table 1.1, and the view of the patrolling system is shown in Figures 1.6 and 1.7.

As the shapes of the tracks increase from hexagon to octagon and 16-gon, there were fewer evasive action maneuvers required when attempting to enter the outer track. This is due to the increased number

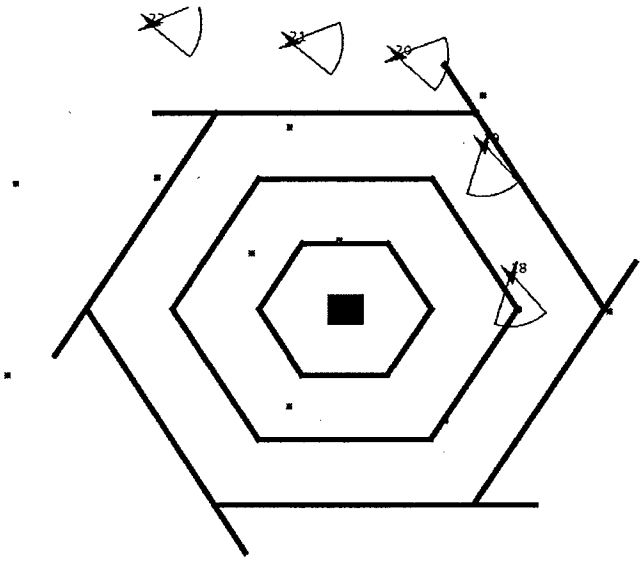

Figure 1.6: View of patrolling simulation, entering patrol.

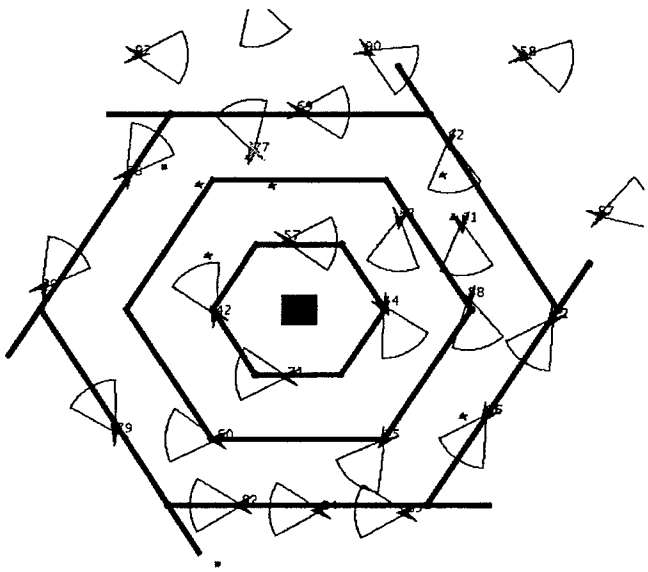

Figure 1.7: View of patrolling simulation, patrolling.

Table 1.1: System performance under varying conditions using hexagonal tracks.

Shape of track	Number of UAVs	Threat density	System performance
Hexagon	1~5	None	4 to 5 UAVs populate the inner most track
		Low	UAVs strike threats before tracks are populated
		High	(Same as above)
	16	None	UAVs populate inner two tracks, no evasive action required
		Low	UAVs populate inner two tracks, most threats destroyed, no evasive action required
		High	UAVs populate inner most track, most threats destroyed, no evasive action required
	32	None	UAVs populate all the three tracks, evasive action required
		Low	UAVs populate all tracks, most threats destroyed, evasive action required
		High	(Same as above)
	> 32	None	UAVs populate all tracks, evasive action required, collisions occurred
		Low	UAVs populate all tracks, most threats destroyed, evasive action required, collisions occurred
		High	UAVs populate all tracks, most threats destroyed, evasive action required

of entry points making it less likely that two UAVs would seek the same entry point simultaneously. With 32-gon tracks, there were more evasive action maneuvers required when attempting to enter the outer track. This is due to the entry points being too close together. The system becomes unstable due to cascading evasive action maneuvers when more than 32 UAVs are in the patrol area.

6. Conclusions and future works

The asset patrol and protection model is implemented and successfully demonstrated in an agent-based, simulated urban environment. The simulation establishes that the emergent, behavior-based patrol procedure for UAVs is effective, robust, and scalable. The approach is especially well suited for numerous, small, inexpensive, and expendable

UAVs. The use of virtual beacons (waypoints), signal-based communication, and simple rules provide a robust and effective method for cooperative control among n UAVs to patrol an asset. The model presented demonstrates that neither high-level control nor high-bandwidth communication is necessary for this complex cooperative control task. The simulation shows that communication is not necessary if all the agents follow the prescribed protocols.

There are several areas that are being explored to expand and extend our current multi-agent model. High level decision layers, based on a Partially Observable Markov Decision Process (POMDP) and a Bayesian Network, are under development to function on the top of the reactive behavior-based agent control. This would allow agents to function more intelligently if more global information is available. In the absence of this global information, agents can fall back on the reactive behavior-based control. The agents may be augmented with a greater behavioral repertoire allowing them to perform a variety of tactics as well as other coordinated movements.

References

[1] [1] Schlecht J, Altenburg K, Ahmed BM, Nygard KE., "Decentralized Search by Unmanned Air Vehicles using Local Communication", Las Vegas, NV: *Proceedings of the International Conference on Artificial Intelligence*, Volume II, pages 757-762, 2003.

[2] Altenburg K, Schlecht J, Nygard KE., "An Agent-based Simulation for Modeling Intelligent Munitions", Athens, Greece: *Advances in Communications and Software Technologies*, pages 60-65, 2002.

POMDP Planning for High Level UAV Decisions: Search vs. Strike

Doug Schesvold, Jingpeng Tang, Benzir Md Ahmed, Karl Altenburg, Kendall E. Nygard

Department of Computer Science and Operations Research
North Dakota State University
Fargo, ND 58105, USA
Kendall.Nygard@ndsu.nodak.edu

Abstract

The Partially Observable Markov Decision Process (POMDP) model is explored for high level decision making for Unmanned Air Vehicles (UAVs). The type of UAV modeled is a flying munition with a limited fuel supply. The UAV is destroyed when it strikes a target. When a UAV detects a target, a decision has to be made whether to continue to search for a better target or strike the current target immediately. Many factors influence this decision, including target value, target density, threat levels, and fuel level. POMDP is a suitable model for this battle field situation because of uncertainties due to the stochastic nature of the problem and the imperfect sensors of the UAV. Two POMDP models are presented in this paper. One uses planning horizon to model the fuel level, while the other models the fuel level explicitly in the states.

1 Introduction

The POMDP models proposed in this paper addresses the Search vs. Strike scenario. The scenario consists of a swarm of unmanned air vehicles (UAVs) that search a given area for specified targets to destroy [1]. Given a rectangular search area, the swarm of UAVs flies in a parallel formation in such a way as to ensure complete coverage of the search area. The UAVs use radar-type sensors to scan for potential targets. The UAVs make as many passes across the search area as needed until the entire area is searched. Each UAV maintains its own private list of detected targets. Once a target is detected, a Search vs. Strike decision must be made. In the real battle field, the sensor information is noisy and imperfect. POMDP explicitly models the imperfect information. Fuel level is an important factor that influences the behavior of the UAVs. With the limited fuel, the UAV must accomplish its mission. A high fuel level makes it more attractive for the UAV to search for a better target, while low fuel level makes it more attractive to strike the best target found. There are two POMDP models presented in this paper. One uses planning horizon to model the fuel level, while the other models the fuel level explicitly in the states.

2 Related Work

POMDP modeling has been used in various areas, such as machine maintenance, medical diagnosis, autonomous robots, and many others. POMDP is used for autonomous office navigation [3] enabling a robot to utilize all its sensor information, for hallway robot navigation problem [4] using function-approximation methods for representing the value functions. POMDP also proves its robustness in modeling problems like searching for a moving target [5]. The Linear Programming and POMDP marriage technique [6] is used for solving large-scale allocation problems with partially observable states and constrained action and observation resources. The POMDP framework is also used in medical applications, such as the management of ischemic heart disease [8].

3 POMDP Model

POMDP extends the Markov Decision Process (MDP) model [1] by allowing partial observability of the system state. The POMDP model is defined by a 6-tuple (S, A, Z, T, R, O). For any system, it defines a finite set of states of the system, S; a finite set of actions, A; transition function, $T(s',s,a) = P(s'|s,a)$, probability of transitioning from one state s to another state s' after taking an action a in state s; rewards, $R(s,a)$, for taking an action a in state s; a finite set of observations, Z; observation function, $O(s,a,z) = P(z|s,a)$, probability of getting observation z given the resulting state s and action a taken in previous state. It may also consist of the horizon length h, the discount factor γ, and the initial belief state. Belief state is the probability distribution over the states, which is kept to access the current system state. In a POMDP model, the belief state has to be updated from the previous belief state. Hence, the process of maintaining the belief state is Markovian. This means that the POMDP model can be seen as a continuous space MDP as the belief space is continuous. The recursive value iteration algorithm for the MDP model can be used to solve the continuous space MDP to find the optimal policy after a little adaptation. More detailed description of the POMDP model can be found in [6].

4 POMDP Modeling of UAV for Search vs. Strike Decision

The first step in constructing a POMDP model of UAV activity is to divide the search area into a grid. A UAV searches one grid element at each time step. The goal is to destroy the highest valued target. Once at least one target is discovered, a decision must be made at each successive grid element of whether to continue to search or to strike the highest value target found thus far. Assume also that the targets degrade in value according to the time elapsed since their discovery. Furthermore, threats may be encountered which may destroy the UAV. The tradeoffs of the search vs. strike decision are illustrated in the following example. Suppose a medium value target is found early on in the mission. The UAV may decide to forgo the certain reward of striking immediately, in hopes of finding a higher value target. The risk associated with making this decision is that the UAV may subsequently only find targets of low value. Also, in the time it took to search for better targets, the medium value target has degraded to a low value target. Thus, the *continue to search* decision cost the UAV the difference between a medium and low value target. However, if the UAV had subsequently found and then struck a higher value target, the decision would have paid off. Another risk associated with the *continue to search* decision is that the UAV may be destroyed by the enemy before it can destroy a target.

4.1 POMDP Formulation

The set of states is:
$S = \{V_iT_j, D \mid 0 \le i \le m, 0 \le j \le n\}$
Where,

V_i: current highest value target
T_j: current highest threat level
D: absorbing state
m, n: number of distinct target and threat values respectively

The highest value may reflect either a newly discovered target, or one of time degraded value. We are assuming that the target value degrades by one for each time step. The state D is an absorbing state corresponding to striking a target, or being shot down by enemy threat. The set of actions is:
$A = \{Strike, Search\}$
The *Strike* action means striking the highest value target found thus far. The *Search* action means searching the next grid element.

The set of observations is:
$O = \{V_iT_j, D \mid 0 \le i \le m, 0 \le j \le n\}$
Where,

V_i: current highest value target
T_j: current highest threat level

The observations correspond directly to the states.

4.1.1 Probability Definitions for the Model

Assume each type (value) of target is equally likely to be found in each grid square. Also assume the value of a discovered target decreases by 1 for each time step. Threats of a given level are also equally likely to be encountered in each grid square. In this model, we assume that the threat level doesn't decay.

Define pV_iT_j $(0 = i = m, 0 = j = n)$ to be the probability of target of value i being highest and threat j being highest in a particular grid square. These are mutually exclusive and they sum to 1. pK_j is the probability of UAV being destroyed by threat j for each time step.

4.1.2 State Transitions

Equations (1) to (5) describe the transition probabilities of the model.

$$P(D \mid V_iT_j, Strike) = 1 \quad (1)$$

$$P(D \mid V_iT_0, Search) = 0 \quad (2)$$

$$P(V_kT_l \mid V_iT_0, Search) = \sum_{valid\ m,n} pV_mT_n \quad (3)$$

$$P(D \mid V_iT_j, Search) = pK_j \quad (4)$$

$$P(V_kT_l \mid V_iT_j, Search) = (1 - pK_j) \sum_{valid\ m,n} pV_mT_n \quad (5)$$

For example, assuming $m = 4$, and $n = 1$, the probability of transitioning from state V_3T_1 to state V_2T_1 is given by $(pV_0T_0 + pV_1T_0 + pV_2T_0 + pV_0T_1 + pV_1T_1 + pV_2T_1) * (1 - pK_1)$. The term pV_3T_1 is not valid because the new state would have the highest value of 3 instead of 2.

4.1.3 Observation Probabilities

If sensors give perfect observations, Equation (6) and (7) describe the observation probabilities.

$$P(D \mid D, Strike) = 1 \quad (6)$$

$$P(V_iT_j \mid V_kT_l, Search) = \begin{cases} 1, & if\ i = k, j = l \\ 0, & otherwise \end{cases} \quad (7\text{-}1)$$

For imperfect observation,

$$P(V_iT_j \mid V_kT_l, Search) = p, 0 \le p \le 1 \quad (7\text{-}2)$$

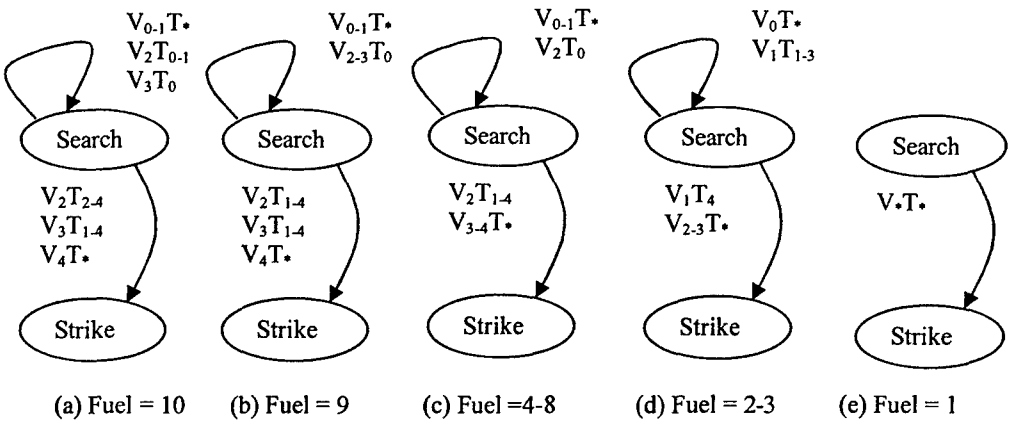

Figure 1. Optimal policy

4.1.4 Reward Function

The reward function is such that no immediate reward is given for the *Search* action. For the *Strike* action, a reward is given that corresponds to the target value of the current state. Equation (8) and (9) describe the reward function.

$$R(V_iT_j, Search) = 0 \quad (8)$$

$$R(V_iT_j, Strike) = i \quad (9)$$

4.1.5 Results

We used POMDP solver written by Anthony Cassandra [????] to solve our POMDP formulations. The exact solver produces output files corresponding to the value function and policy graph of the solution. After experimentation with several data sets modeling imperfect sensors, it was discovered that the exact solver could not produce solutions in a reasonable amount of time. Therefore, we present results for the model that assumes perfect sensors. This reduces the POMDP problem to an MDP. Since modeling imperfect sensors is of high interest, future work will involve the use of heuristic POMDP solvers.

The use of the model is demonstrated with an example. The targets and threats are modeled with the following probabilities for a grid square:

Probability of target values (0-4) being highest: 0.37, 0.19, 0.17, 0.15, 0.12

Probability of threat levels (0-4) being highest: 0.95, 0.02, 0.02, 0.005, 0.005

The ViTj probabilities are easily calculated from the above probabilities assuming independence of targets and threats.

Probability of being shot down per time step from threat level (0-4): 0.0, 0.16, 0.2, 0.25, 0.33

Figure 4.1 shows the optimal policy for the given input values. Subscript notation can include ranges and wildcards. The subscript values indicate the observation of target value (V) and threat level (T).

5 POMDP Model with Fuel State Information

Based on the previous model, fuel level is introduced in this model. The system state is described by the combination of the target value, threat level, and fuel level. The state describes two aspects of the environment: external (target value, threat level) and internal (fuel level).

The POMDP model is described as follow:

(1) action : strike, search

(2) states :

$S = \{V_iT_jF_k, D \mid 0=i=m, 0=j=n, 1=k=q\}$

Where,

V_i: current highest value target

T_j: current highest threat level

F_k: current fuel level

D: absorbing state

m, n, q: number of distinct target, threat and fuel values respectively

(3) observation: $V_iT_jF_k$

(4) transition function: the only legal transition for fuel level is transfer to one level lower state or stay in the same fuel level state.

(5) observation function:

$$P(V_i T_j F_k \mid V_l T_m F_n, Search) = p, 0 \le p \le 1 \quad (10)$$

The policy graph shown in Figure 5.1 describes the action taken by the UAV based on different observations. For this example, there are 2×2×2 = 8 possible observations. From the figure we see that the UAV continues searching if the fuel level is high or the target value is low. The UAV will strike if the target value is high and the fuel level is low. Assignment of reward values and target values are arbitrary in our work. In practice, the assignment would reflect the professional judgment of some military officer. The improvement of this model is that fuel level is explicitly described, which is an important feature that reflects the short life of UAV.

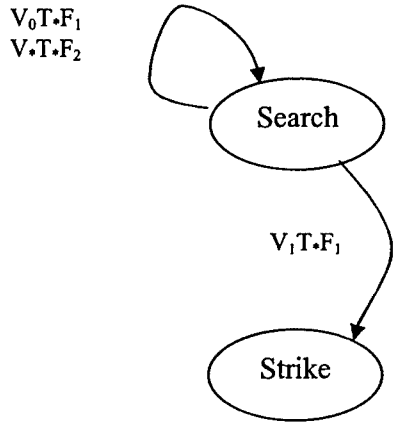

Figure 2. Optimal policy for explicit fuel level

6 Conclusions and Future Work

The POMDP model is explored for high level decision making for Unmanned Air Vehicles (UAVs). Reasonable policies for strike vs search are obtained by the POMDP model. In general, the POMDP model is not scalable. Solutions for all but small problems are intractable. Only problems with very few states and actions can be solved exactly. However, a good heuristic is needed to solve POMDP model for problems of moderate size in reasonable time. Heuristic methods for POMDP is an area of active research

For the scalability problem, to get a policy online fast, building a policy library off-line may be a feasible solution. The UAV can easily and quickly find the suitable policy searching existing policies based on the current situation. Case based reasoning techniques, such as RABIT (Retrieval with Attribute Based Indexing Technique) [9-10], can be used in tracing the policy. The problem with the offline solution is, policy library may not completely reflect all real battle field situations. A better solution may be to use newly acquired information to improve the best possible offline policy heuristically. A good general purpose heuristic solution is yet to be discovered. One main limitation in using POMDP model is that accurate information required by the POMDP model may not be readily available.

References

[1] Joseph Schlecht, Karl Altenburg, Benzir Md Ahmed, Kendall E. Nygard, Decentralized Search by Unmanned Air Vehicles using Local Communication

[2] Howard, R. A. 1960. *Dynamic Programming and Markov Processes*. MIT Press.

[3] Simmons, R., and Koenig, S. 1995. *Probabilistic navigation in partially observable environments.* Fourteenth International Joint Conference on Artificial Intelligence, 1080--1087. Montreal, Canada: Morgan Kaufmann.

[4] Cassandra, A.; Kaelbling, L.; and Kurien, J. 1996. *Acting under uncertainty: Discrete bayesian models for mobile-robot navigation.* Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems.

[5] Eagle, J.N., *The Optimal Search for a Moving Target When the Search Path is Constrained*, Operations Research, Vol. 32, No. 5, September-October 1984, pp. 1107-1115.

[6] Yost, K. and Washburn, A.R., *The LP/POMDP Marriage: Optimization with Imperfect Information*, Naval Research Logistics, Vol. 47, No. 8, 2000, pp. 607-619.

[7] Cassandra, A. R. 1998. Exact and Approximate Algorithms for Partially Observable Markov Decision Processes. Ph.D. Dissertation, Department of Computer Science, Brown University.

[8] M. Hauskrecht. Dynamic decision making in stochastic partially observable medical domains: ischemic heart disease example, Proceedings of AI in Medicine Europe (AIME), Grenoble, France, pp. 296-299, 1997.

[http://www.cs.brown.edu/research/ai/pomdp/]

[9] Wolfgang Grather 1994. Computing Distances between attribute-value reorientations in an associative memory. Similarity Cocepts and Retrieval Methods, volume 13 of Fabel-Report, pages 12-25. GMD, Sankt Augustin, 1994.

[10] Bernd Linowski. RABIT: Ein objektorientiertes System zum Retrieval attrbutbasierter falle. Febel-Report 34, GND, Sankt Augustin, June 1995, 162 pages.